ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

УПРАВЛЕНИЕ ЦИФРОВЫХ ОБРАЗОВАТЕЛЬНЫХ ТЕХНОЛОГИЙ

Кафедра «Автоматизация производственных процессов»

Практикум

«Matlab, базовые понятия и функции»
по дисциплине

«Пакеты прикладных программ»

Авторы
Туркин И. А.,
Носачев С. В.,
Зимовнов О. В.,
Лапшин В. П.

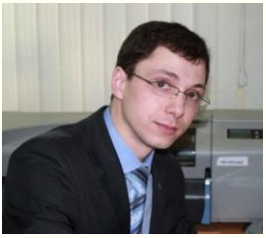
Ростов-на-Дону, 2019

Аннотация

Практикум предназначен для студентов очной, заочной форм обучения направлений 27.03.04 «Управление в технических системах», 15.03.04 «Автоматизация технологических процессов и производств».

Авторы

к.т.н, доцент,
каф. АПП
Туркин И.А.



к.т.н., доцент,
каф. АПП
Носачев С.В.

к.т.н., доцент, доцент,
каф. АПП
Зимовнов О.В.

к.т.н., доцент, доцент,
каф. АПП
Лапшин В.П.

Оглавление

ВВЕДЕНИЕ

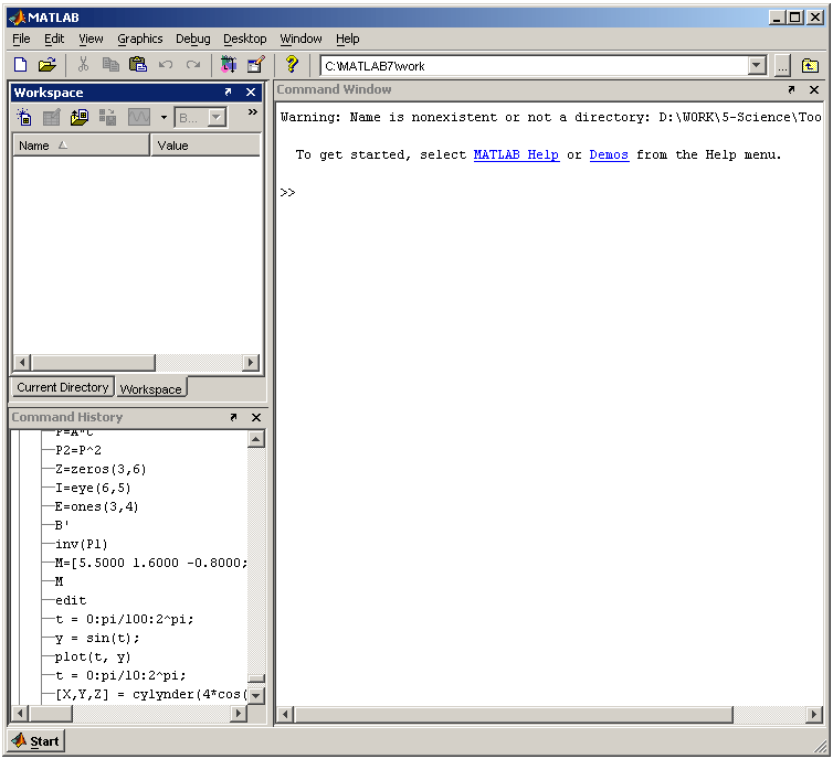
В этом пособии рассказывается о применении пакета MatLab для анализа многомерных данных. В пособии интенсивно используются понятия и методы матричной алгебры – вектор, матрица, и т.п. MatLab является мощным и универсальным средством обработки многомерных данных. Сама структура пакета делает его удобным средством для проведения матричных вычислений. Спектр проблем, исследование которых может, осуществлено при помощи MatLab, охватывает: матричный анализ, обработку сигналов и изображений, нейронные сети и многие другие. MatLab — это язык высокого уровня, имеющий открытый код, что дает возможность опытным пользователям разбираться в запрограммированных алгоритмах. Простой встроенный язык программирования позволяет легко создавать собственные алгоритмы. За много лет использования MatLab создано огромное количество функций и ToolBox (пакетов специализированных средств), которые значительно расширяют функционал пакета.

1. БАЗОВЫЕ СВЕДЕНИЯ

1.1. Рабочая среда MatLab

Чтобы запустить программу дважды щелкните на иконку

Перед Вами откроется рабочая среда, изображенная на рисунке.



Рабочая среда MatLab 6.x немного отличается от рабочей среды предыдущих версий, она имеет более удобный интерфейс для доступа ко многим вспомогательным элементам

Рабочая среда MatLab 6.x содержит следующие элементы:

панель инструментов с кнопками и раскрывающимся списком;

окно с вкладками **Launch Pad** и **Workspace**, из которого можно получить доступ к различным модулям ToolBox и к содержимому рабочей среды;

окно с вкладками **Command History** и **Current Directory**, предназначенное для просмотра и повторного вызова ранее введенных команд, а также для установки текущего каталога;

командное окно, в котором находится приглашение к вводу » и мигающий вертикальный курсор;

строку состояния.

Если в рабочей среде MatLab 6.x отсутствуют некоторые окна, приведенные на рисунке, то следует в меню **View** выбрать

соответствующие пункты: **Command Window**, **Command History**, **Current Directory**, **Workspase**, **Launch Pad**.

Команды следует набирать в командном окне. Символ », обозначающий приглашение к вводу командной строки, набирать не нужно. Для просмотра рабочей области удобно использовать полосы скроллинга или клавиши **Home**, **End**, для перемещения влево или вправо, и **PageUp**, **PageDown** для перемещения вверх или вниз. Если вдруг после перемещения по рабочей области командного окна пропала командная строка с мигающим курсором, просто нажмите **Enter**.

Важно помнить, что набор любой команды или выражения должен заканчиваться нажатием на **Enter**, для того, чтобы программа MatLab выполнила эту команду или вычислила выражение.

1.2. Простейшие вычисления

Наберите в командной строке `1+2` и нажмите **Enter**. В результате в командном окне MatLab отображается следующее:

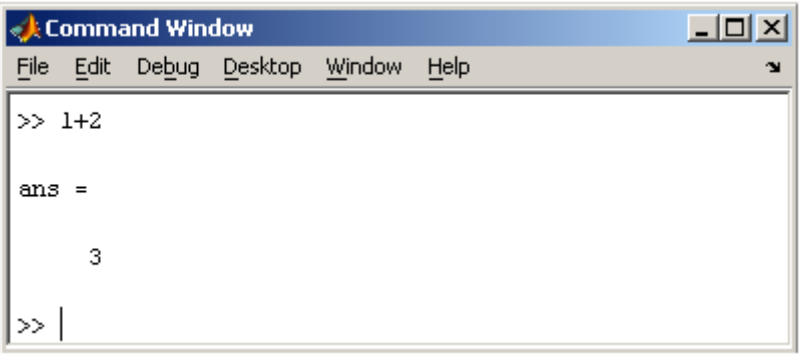


Рис. 2 Графическое представление метода главных компонент

Что сделала программа MatLab? Сначала она вычислила сумму `1+2`, затем записала результат в специальную переменную `ans` и вывела ее значение, равное `3`, в командное окно. Ниже ответа расположена командная строка с мигающим курсором, обозначающая, что MatLab готов к дальнейшим вычислениям. Можно набирать в командной строке новые выражения и находить их значения. Если требуется продолжить работу с предыдущим выражением, например, вычислить `(1+2)/4.5`, то проще всего воспользоваться уже имеющимся результатом, который хранится в переменной `ans`. Наберите `ans/4.5` (при вводе десятичных дро-

бей используется точка) и нажмите **Enter**, получается

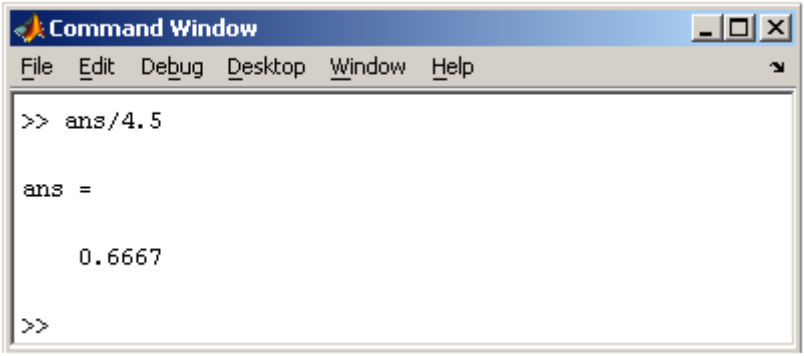


Рис. 3 Графическое представление метода главных компонент

1.3. Эхо команд

Выполнение каждой команды в MatLab сопровождается эхом. В приведенном выше примере — это ответ `ans = 0.6667`. Часто эхо затрудняет восприятие работы программы и тогда его можно отключить. Для этого команда должна завершаться символом точка с запятой. Например

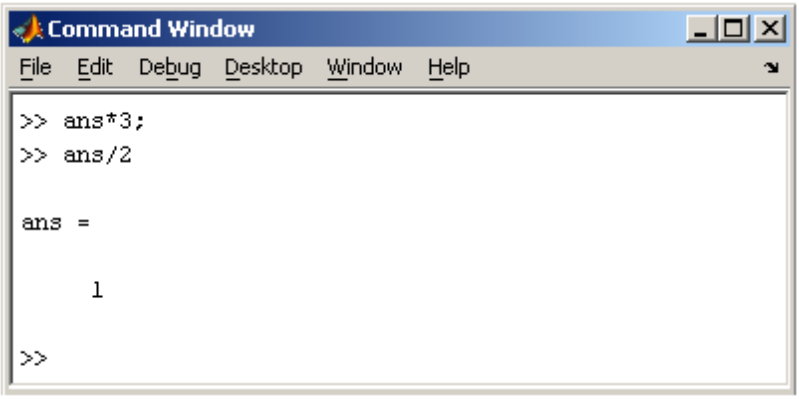


Рис. 4 Пример ввода функции ScoresPCA

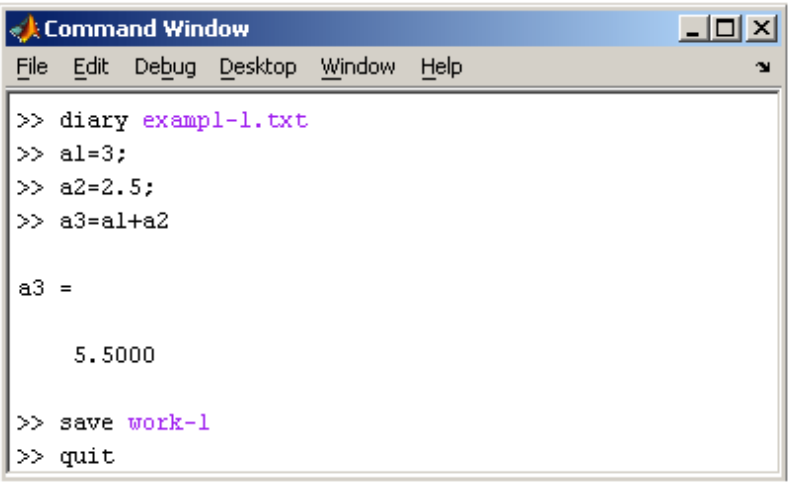
1.4. Сохранение рабочей среды. MAT файлы

Самый простой способ сохранить все значения переменных — использовать в меню **File** пункт **Save Workspase As.** При этом появляется диалоговое окно **Save Workspase Variables**, в котором следует указать каталог и имя файла. По умолчанию предлагается сохранить файл в подкаталоге `work` основного каталога MatLab. Программа со- хранит результаты работы в

файле с расширением `mat`. Теперь можно закрыть MatLab. В следующем сеансе работы для восстановления значений переменных следует открыть этот сохраненный файл при помощи подпункта **`Open`** меню **`File`**. Теперь все переменные, определенные в прошлом сеансе, опять стали доступными. Их можно использовать во вновь вводимых командах.

1.5. Журнал

В MatLab имеется возможность записывать исполняемые команды и результаты в текстовый файл (вести журнал работы), который потом можно прочитать или распечатать из текстового редактора. Для начала ведения журнала служит команда `diary`. В качестве аргумента команды `diary` следует задать имя файла, в котором будет храниться журнал работы. Набираемые далее команды и результаты их исполнения будут записываться я в этот файл, например последовательность команд



производит следующие действия:

открывает журнал в файле `exampl-1.txt`;

производит вычисления;

сохраняет все переменные в MAT файле `work-1.mat`;

сохраняет журнал в файле `exampl-1.txt` в подкаталоге `work` корневого каталога MatLab и закрывает MatLab;

Посмотрите содержимое файла `exampl-1.txt` в каком-нибудь текстовом редакторе. В файле окажется следующий текст:

```
a1=3;
a2=2.5;
a3=a1+a2

a3 =

    5.5000

save work-1
quit
```

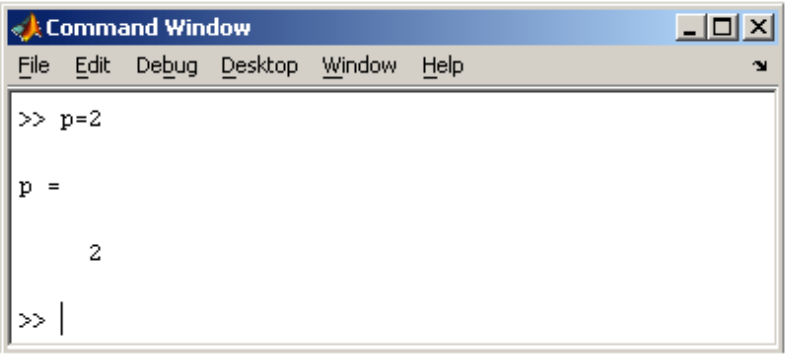
1.6. Система помощи

Окно справки MatLab появляется после выбора опции **`Help Window`** в меню **`Help`** или нажатием кнопки вопроса на панели инструментов. Эта же операция может быть выполнена при наборе команды `helpwin`. Для вывода окна справки по отдельным разделам, наберите `helpwin topic`. Окно справки предоставляет Вам такую же информацию, как и команда `help`, но оконный интерфейс обеспечивает более удобную связь с другими разделами справки.

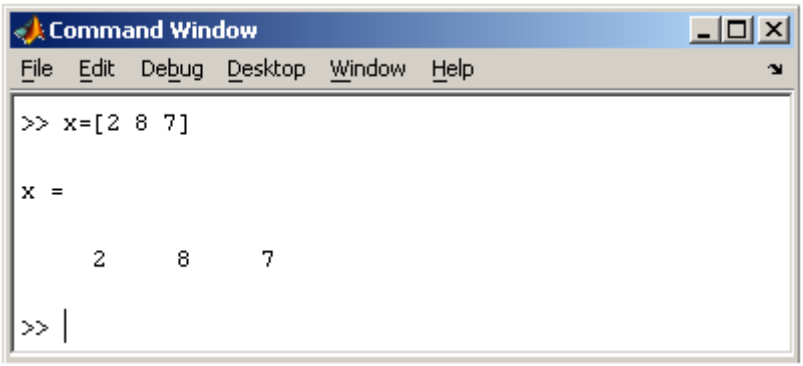
2. МАТРИЦЫ

2.1. Скаляры, векторы и матрицы

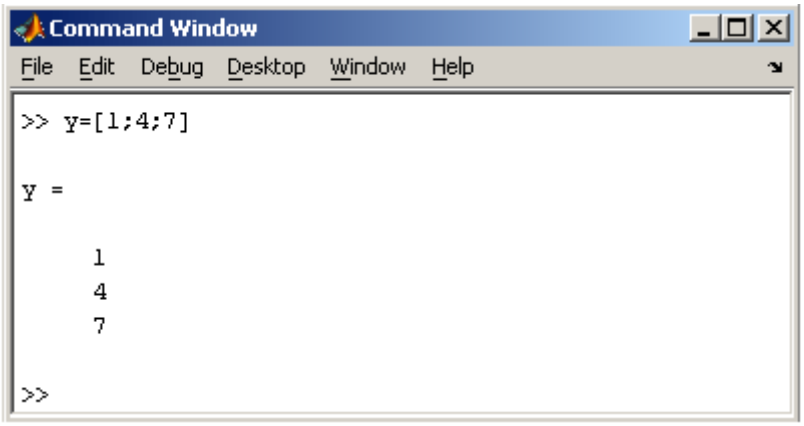
В MatLab можно использовать скаляры, векторы и матрицы. Для ввода скаляра достаточно приписать его значение какой-то переменной, например



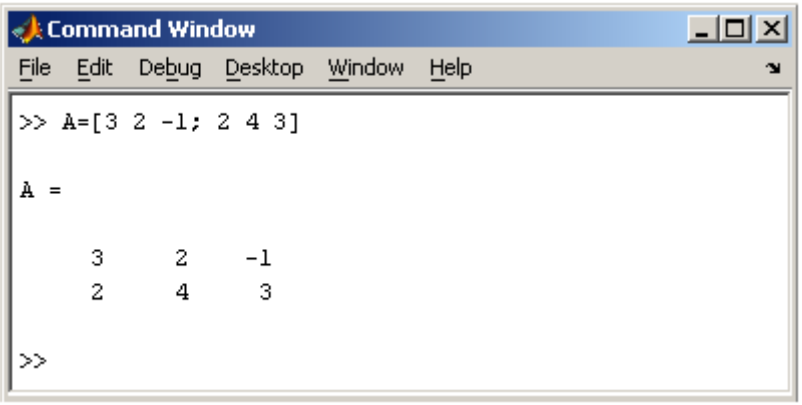
Заметим, что MatLab различает заглавные и прописные буквы, так что p и P — это разные переменные. Для ввода массивов (векторов или матриц) их элементы заключают в квадратные скобки. Так для ввода вектора-строки размером 1×3, используется следующая команда, в которой элементы строки отделяются пробелами или запятыми.



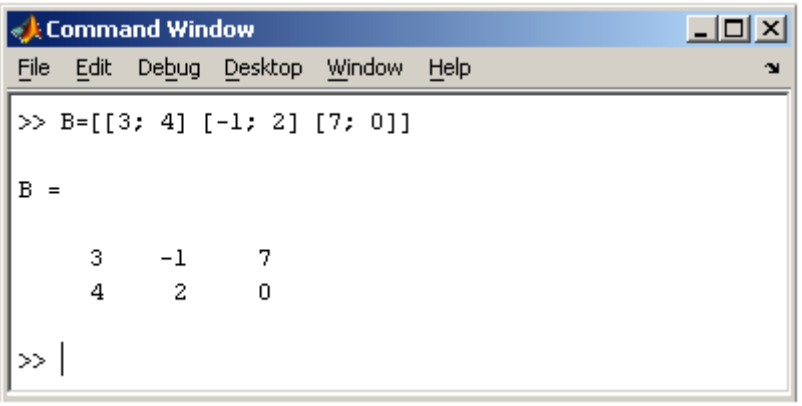
При вводе вектора-столбца элементы разделяют точкой с запятой. Например,



Вводить небольшие по размеру матрицы удобно прямо из командной строки. При вводе матрицу можно рассматривать как вектор-столбец, каждый элемент которого является вектор-строкой.

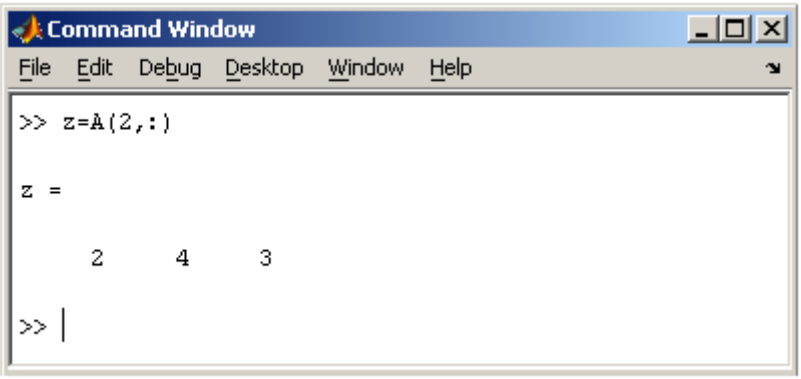


или матрицу можно трактовать как вектор строку, каждый элемент которой является вектор-столбцом.

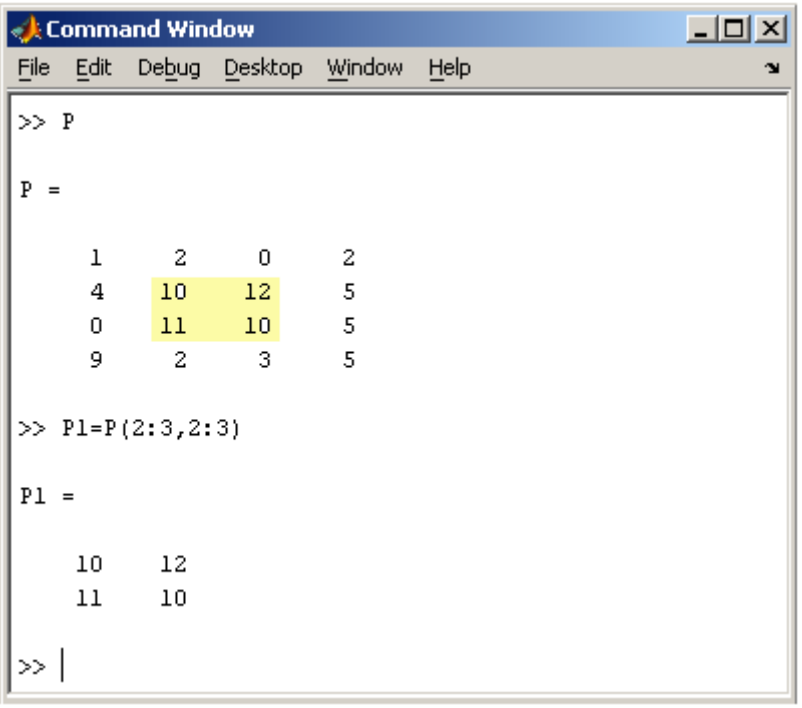


2.2. Доступ к элементам

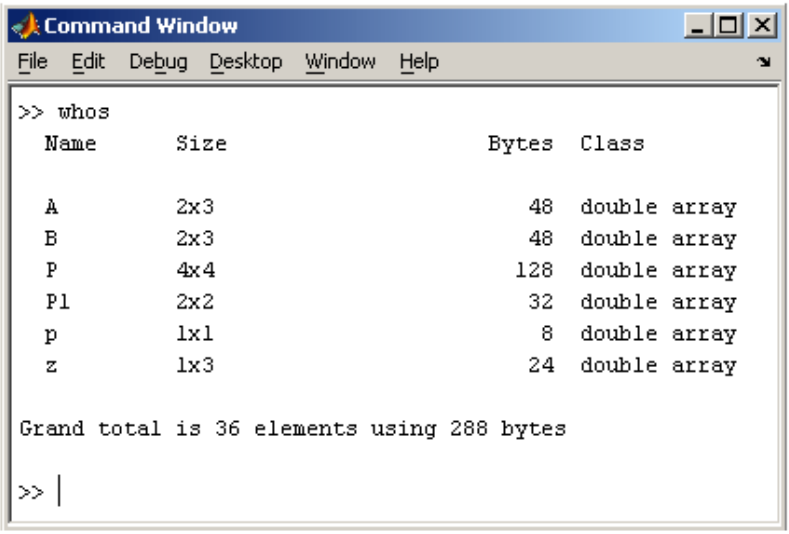
Доступ к элементам матриц осуществляется при помощи двух индексов — номеров строки и столбца, заключенных в круглые скобки, например команда `B(2,3)` выдаст элемент второй строки и третьего столбца матрицы `B`. Для выделения из матрицы столбца или строки следует в качестве одного из индексов использовать номер столбца или строки матрицы, а другой индекс заменить двоеточием. Например, запишем вторую строку матрицы `A` в вектор `z`



Также можно осуществлять выделение блоков матриц при помощи двоеточия. Например, выделим из матрицы `P` блок отмеченный цветом



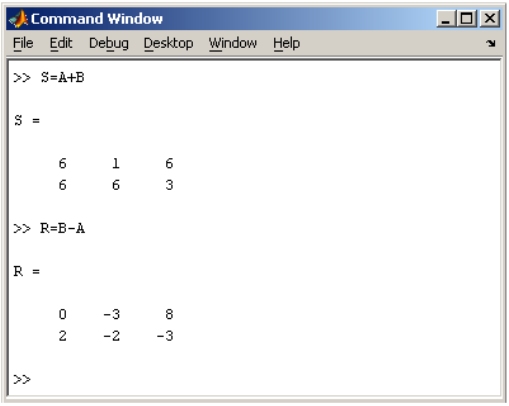
Если необходимо посмотреть переменные рабочей среды, в командной строке необходимо набрать команду `whos`.



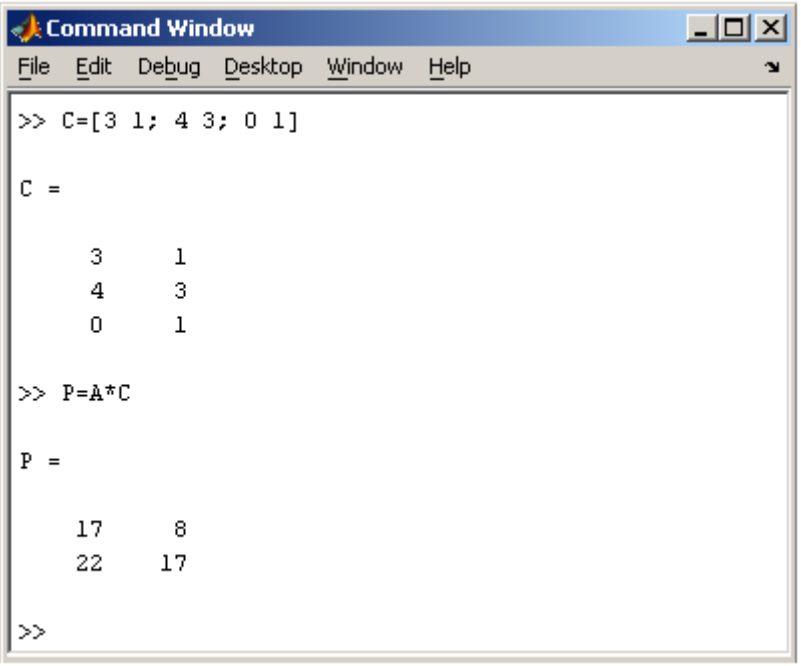
Видно, что в рабочей среде содержатся один скаляр (p), четыре матрицы (A, B, P, P1) и вектор-строка (z).

2.3. Основные матричные операции

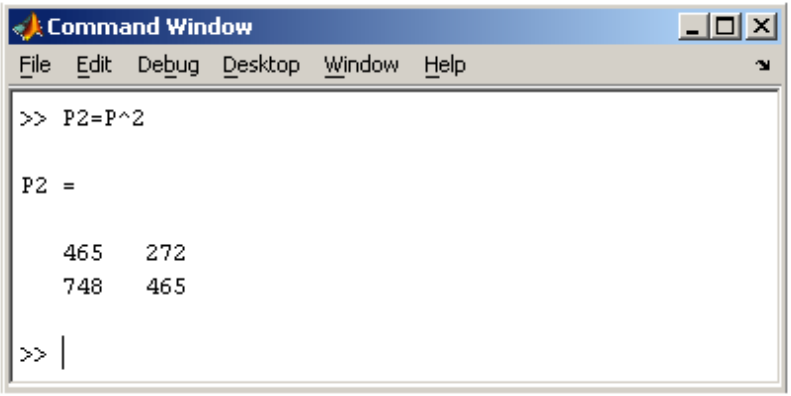
При использовании матричных операций следует помнить, что для сложения или вычитания матрицы должны быть одного размера, а при перемножении число столбцов первой матрицы обязано равняться числу строк второй матрицы. Сложение и вычитание матриц, так же как чисел и векторов, осуществляется при помощи знаков плюс и минус



а умножение — знаком звездочка *. Введем матрицу размером 3×2



Умножение матрицы на число тоже осуществляется при помощи звездочки, причем умножать на число можно как справа, так и слева. Возведение квадратной матрицы в целую степень производится с использованием оператора ^

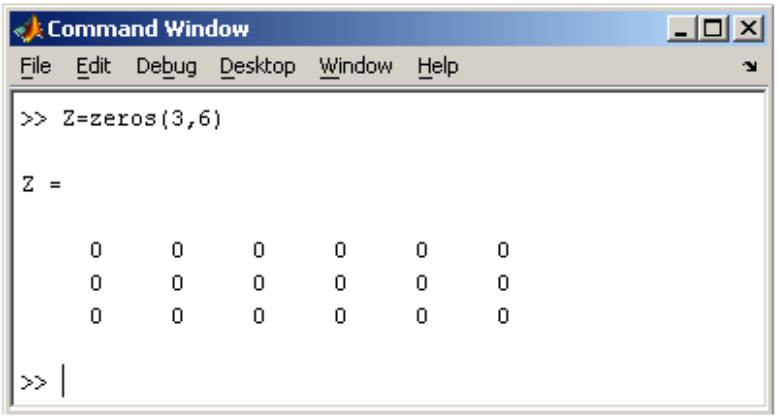


Проверьте полученный результат, умножив матрицу P саму

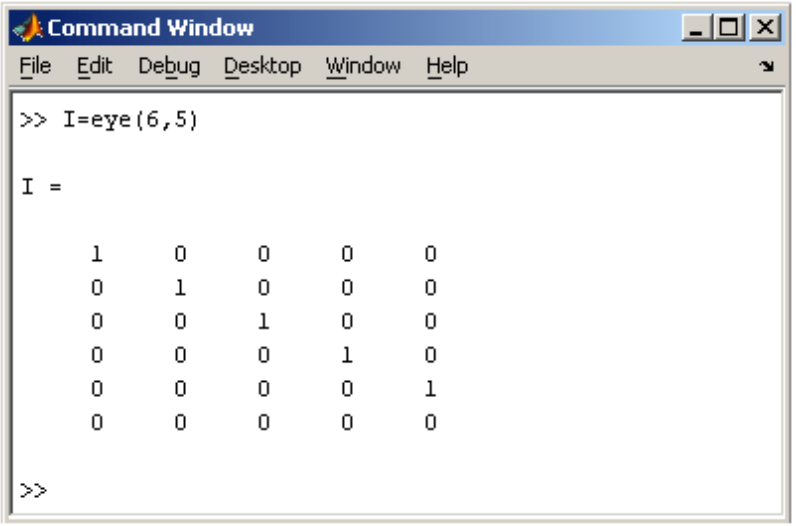
на себя.

2.4. Создание матриц специального вида

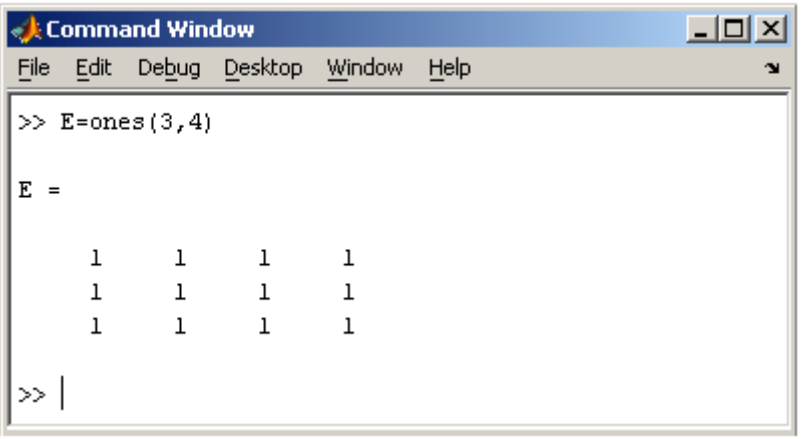
Заполнение прямоугольной матрицы нулями производится встроенной функцией `zeros`



Единичная матрица создается при помощи функции `eye`



Матрица, состоящая из единиц, образуется в результате вызова функции `ones`

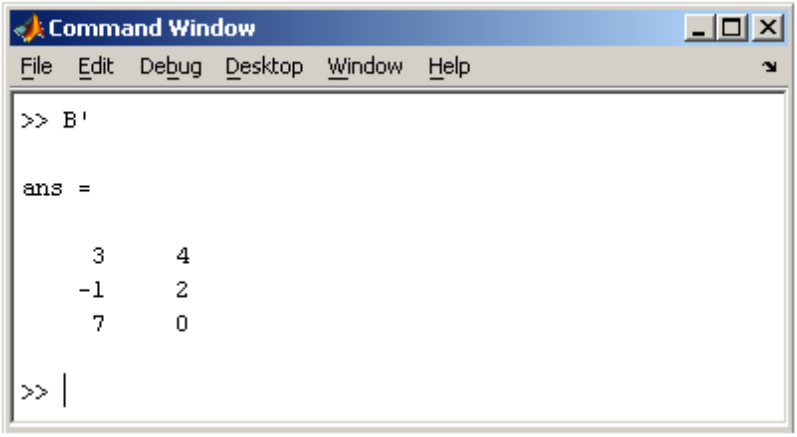


MatLab предоставляет возможность заполнения матриц случайными числами. Результатом функции `rand` является матрица чисел, равномерно распределенных между нулем и единицей, а функции `randn` — матрица чисел, распределенных по нормальному закону с нулевым средним и единичной дисперсией.

Функция `diag` формирует диагональную матрицу из вектора, располагая элементы по диагонали.

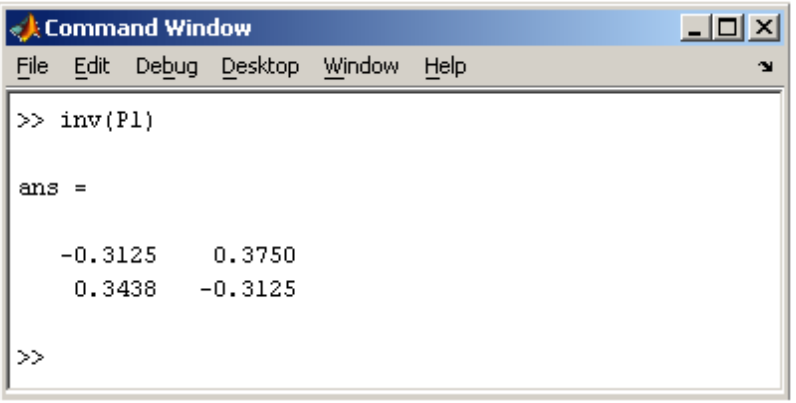
2.5. Матричные вычисления

MatLab содержит множество различных функций для работы с матрицами. Так, например, транспонирование матрицы производится при помощи апострофа `'`



Нахождение обратной матрицы проводится с помощью

функции `inv` для квадратных матриц



Псевдообратную матрицу можно найти с помощью функции `pinv`.

Более подробно про обработку матричных данных можно узнать, если вывести список всех встроенных функций обработки данных командой `help datafun`, а затем посмотреть информацию о нужной функции, например `help max`.

2. ИНТЕГРИРОВАНИЕ MATLAB И EXCEL

Интегрирование MatLab и Excel позволяет пользователю Excel обращаться к многочисленным функциям MatLab для обработки данных, различных вычислений и визуализации результата. Надстройка `excllink.xla` реализует данное расширение возможностей Excel. Для связи MatLab и Excel определены специальные функции.

3.1. Конфигурирование Excel

Перед тем как настраивать Excel на совместную работу с MatLab, следует убедиться, что Excel Link входит в установленную версию MatLab. В подкаталоге `exclink` основного каталога MatLab или подкаталога `toolbox` должен находиться файл с надстройкой `excllink.xla`. Запустите Excel и в меню **`Tools`** выберите пункт **`Add-ins`**. Откроется диалоговое окно, содержащее информацию о доступных в данный момент надстройках. Используя кнопку **`Browse`**, укажите путь к файлу `excllink.xla`. В списке надстроек диалогового окна появтится строка **`Excel Link 2.0 for use with MatLab`** с установленным флагом. Нажмите **`OK`**, требуемая надстройка добавлена в Excel.

Обратите внимание, что в Excel теперь присутствует панель инструментов **Excel Link**, содержащая три кнопки: **putmatrix**, **getmatrix**, **evalstring**. Эти кнопки реализуют основные действия, требуемые для осуществления взаимосвязи между Excel и MatLab — обмен матричными данными, и выполнение команд MatLab из среды Excel. При повторных запусках Excel надстройка `excllink.xla` подключается автоматически.

Согласованная работа Excel и MatLab требует еще нескольких установок, которые приняты в Excel по умолчанию (но могут быть изменены). В меню **Tools** перейдите к пункту **Options**, открывается диалоговое окно **Options**. Выберите вкладку **General** и убедитесь, что флаг **R1C1 reference style** выключен, т.е. ячейки нумеруются A1, A2 и т.д. На вкладке **Edit** должен быть установлен флаг **Move selection after Enter**.

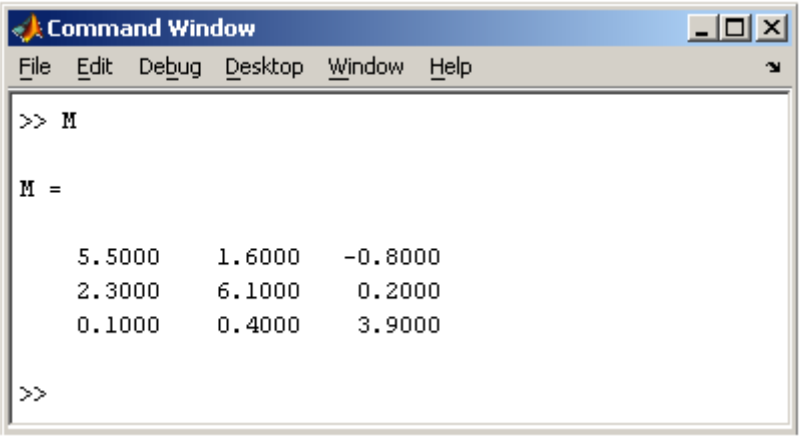
3.2. Обмен данными между MatLab и Excel

Запустите Excel, проверьте, что проделаны все необходимые настройки так, как описано в предыдущем разделе (MatLab должен быть закрыт). Введите в ячейки с A1 по C3 матрицу, для отделения десятичных знаков используйте точку в соответствии с требованиями Excel.

	A	B	C
1	5.5	1.6	-0.8
2	2.3	6.1	0.2
3	0.1	0.4	3.9

Выделите на листе данные ячейки и нажмите кнопку **putmatrix**, появляется окно Excel с предупреждением о том, что MatLab не запущен. Нажмите **OK**, дождитесь открытия MatLab.

Появляется диалоговое окно Excel со строкой ввода, предназначенной для определения имени переменной рабочей среды MatLab, в которую следует экспортировать данные из выделенных ячеек Excel. Введите к примеру, `M` и закройте окно при помощи кнопки **OK**. Перейдите к командному окну MatLab и убедитесь, что в рабочей среде создалась переменная `M`, содержащая массив три на три:



Проделайте некоторые операции в MatLab с матрицей `M`, например, обратите ее.

Вызов `inv` для обращения матрицы, как и любой другой команды MatLab можно осуществить прямо из Excel. Нажатие на кнопку **`evalstring`**, расположенную на панели **`Excel Link`**, приводит к появлению диалогового окна, в строке ввода которого следует набрать команду MatLab

`IM=inv(M)`.

Результат аналогичен полученному при выполнении команды в среде MatLab.

Вернитесь в Excel, сделайте текущей ячейку A5 и нажмите кнопку **`getmatrix`**. Появляется диалоговое окно со строкой ввода, в которой требуется ввести имя переменной, импортируемой в Excel. В данном случае такой переменной является `IM`. Нажмите OK, в ячейки с A5 по A7 введены элементы обратной матрицы.

Итак, для экспорта матрицы в MatLab следует выделить подходящие ячейки листа Excel, а для импорта достаточно указать одну ячейку, которая будет являться верхним левым элементом импортируемого массива. Остальные элементы запишутся в ячейки листа согласно размерам массива, переписывая содержащиеся в них данные, поэтому следует соблюдать осторожность при импорте массивов.

Вышеописанный подход является самым простым способом обмена информацией между приложениями — исходные данные содержатся в Excel, затем экспортируются в MatLab, обрабатываются там некоторым образом и результат импортируется в Excel. Пользователь переносит данные при помощи кнопок панели ин-

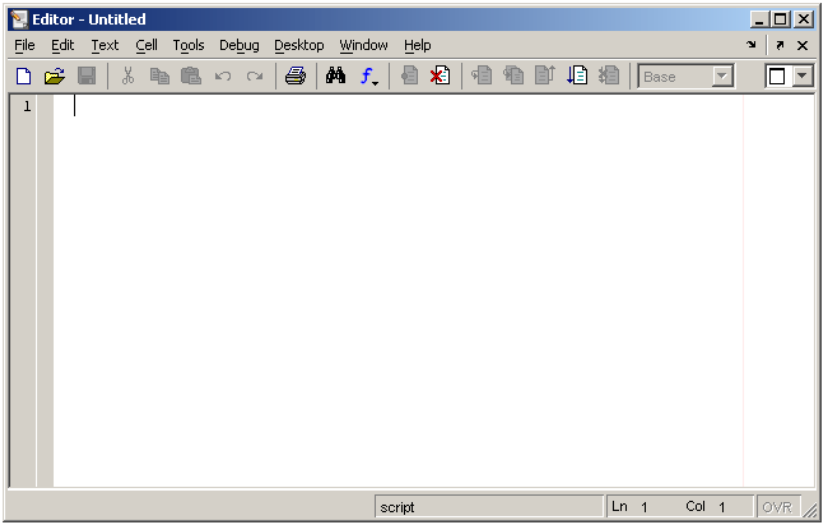
струментов **Excel Link**. Информация может быть представлена в виде матрицы, т.е. прямоугольной области рабочего листа. Ячейки, расположенные в строку или столбец, экспортируются, соответственно, в вектор-строки и вектор-столбцы MatLab. Аналогично происходит и импорт вектор-строк и вектор-столбцов в Excel.

3. ПРОГРАММИРОВАНИЕ

4.1. M-файлы

Работа из командной строки MatLab затрудняется, если требуется вводить много команд и часто их изменять. Ведение дневника при помощи команды `diary` и сохранение рабочей среды незначительно облегчают работу. Самым удобным способом выполнения групп команд MatLab является использование M-файлов, в которых можно набирать команды, выполнять их все сразу или частями, сохранять в файле и использовать в дальнейшем. Для работы с M-файлами предназначен редактор M-файлов. С его помощью можно создавать собственные функции и вызывать их, в том числе и из командного окна.

Раскройте меню **File** основного окна MatLab и в пункте **New** выберите подпункт **M-file**. Новый файл открывается в окне редактора M-файлов, которое изображено на рисунке.

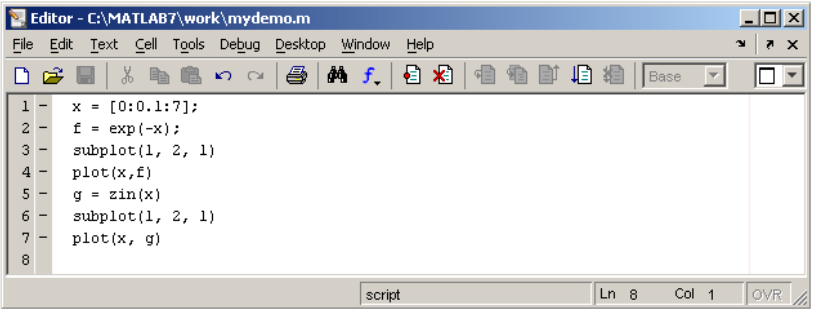


M-файлы в MatLab бывают двух типов: файл-программы (*Script M-Files*), содержащие последовательность команд, и файл-

функции, (*Function M-Files*), в которых описываются функции, определяемые пользователем.

4.2. Файл-программа

Наберите в редакторе команды, приводящие к построению двух графиков на одном графическом окне

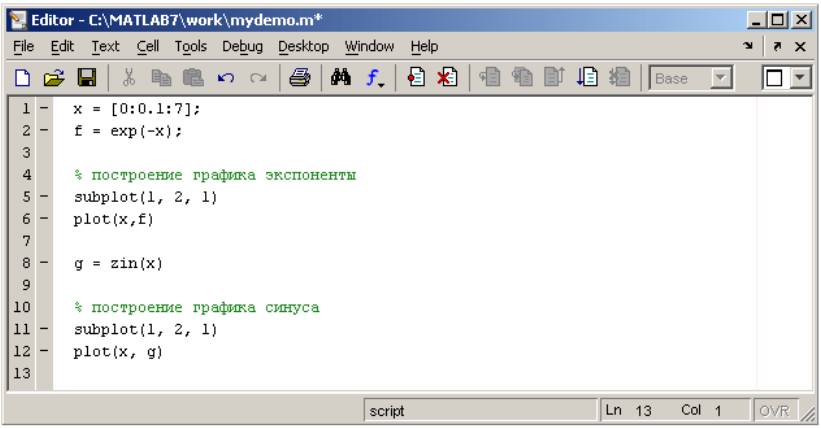


Сохраните теперь файл с именем `mydemo.m` в подкаталоге `work` основного каталога MatLab, выбрав пункт **`Save as`** меню **`File`** редактора. Для запуска на выполнение всех команд, содержащихся в файле, следует выбрать пункт **`Run`** в меню **`Debug`**. На экране появится графическое окно *Figure 1*, содержащее графики функций.

Команды файл-программы осуществляют вывод в командное окно. Для подавления вывода следует завершать команды точкой с запятой. Если при наборе сделана ошибка и MatLab не может распознать команду, то происходит выполнение команд до неправильно введенной, после чего выводится сообщение об ошибки в командное окно.

Очень удобной возможностью, предоставляемой редактором М-файлов, является выполнение части команд. Закройте графическое окно *Figure 1*. Выделите при помощи мыши, удерживая левую кнопку, или клавишами со стрелками при нажатой клавише **`Shift`**, первые четыре команды и выполните их из пункта **`Text`**. Обратите внимание, что в графическое окно вывелся только один график, соответствующий выполненным: командам. Запомните, что для выполнения части команд их следует выделить и нажать клавишу **`F9`**.

Отдельные блоки М-файла можно снабжать комментариями, которые пропускаются при выполнении, но удобны при работе с М-файлом. Комментарии начинаются со знака процента и автоматически выделяются зеленым цветом, например:

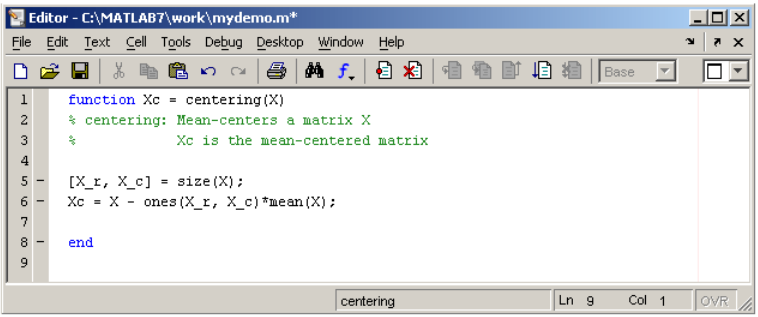


Открытие существующего М-файла производится при помощи пункта **Open** меню **File** рабочей среды, либо редактора М-файлов.

4.3. Файл-функция

Рассмотренная выше файл-программа является только последовательностью команд MatLab, она не имеет входных и выходных аргументов. Для использования численных методов и при программировании собственных приложений в MatLab необходимо уметь составлять файл-функции, которые производят необходимые действия с входными аргументами и возвращают результат действия в выходных аргументах. Разберем несколько простых примеров, позволяющих понять работу с файл-функциями.

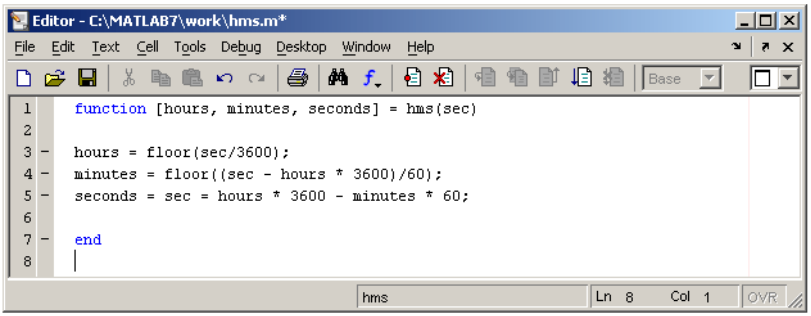
Проводя предобработку данных многомерного анализа хемометрики часто применяет центрирование. Имеет смысл один раз написать файл-функцию, а потом вызывать его всюду, где необходимо производить центрирование. Откройте в редакторе М-файлов новый файл и наберите



Слово `function` в первой строке определяет, что данный файл содержит файл-функцию. Первая строка является заголовком функции, в которой размещается имя функции и списка входных и выходных аргументов. В примере имя функции `centering`, один входной аргумент `X` и один выходной — `Xc`. После заголовка следуют комментарии, а затем — тело функции (оно в данном примере состоит из двух строк), где и вычисляется ее значение. Важно, что вычисленное значение записывается в `Xc`. Не забудьте поставить точку с запятой для предотвращения вывода лишней информации на экран. Теперь сохраните файл в рабочем каталоге. Обратите внимание, что выбор пункта **`Save`** или **`Save as`** меню **`File`** приводит к появлению диалогового окна сохранения файла, в поле **`File name`** которого уже содержится название `centering`. Не изменяйте его, сохраните файл функцию в файле с предложенным именем!

Теперь созданную функцию можно использовать так же, как и встроенные `sin`, `cos` и другие. Вызов собственных функций может осуществляться из файл-программы и из другой файл-функции. Попробуйте сами написать файл-функцию, которая будет шкалировать матрицы, т.е. делить каждый столбец на величину среднеквадратичного отклонения по этому столбцу.

Можно написать файл-функции с несколькими входными аргументами, которые размещаются в списке через запятую. Можно также создавать и функции, возвращающие несколько значений. Для этого выходные аргументы добавляются через запятую в список выходных аргументов, а сам список заключается в квадратные скобки. Хорошим примером является функция, переводящая время, заданное в секундах, в часы, минуты и секунды.

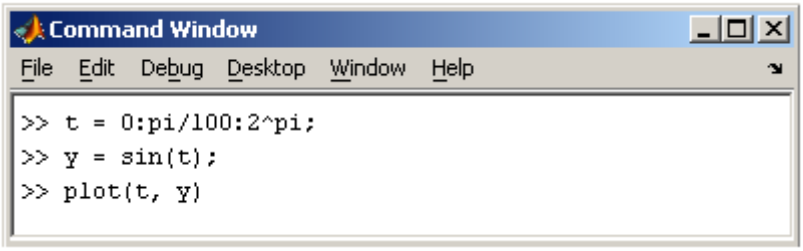


При вызове файл-функций с несколькими выходными аргументами результат следует записывать в вектор соответствующей длины.

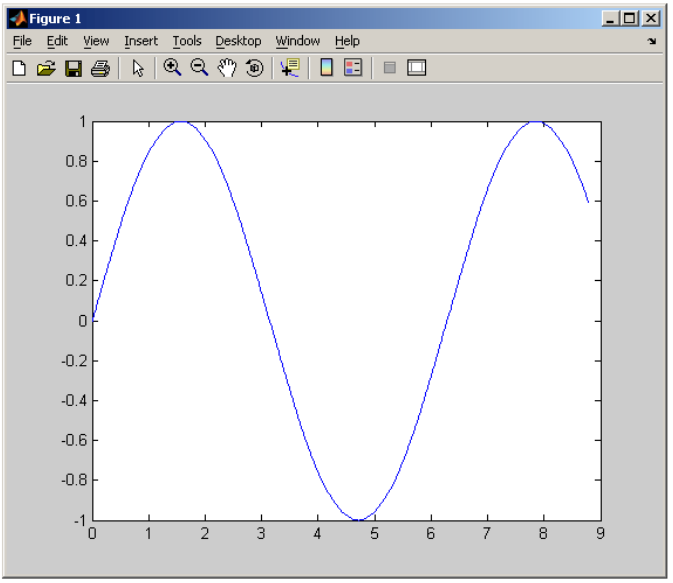
4.4 Создание графика

MatLab имеет широкие возможности для графического изображения векторов и матриц, а также для создания комментариев и печати графиков. Дадим описание несколько важных графических функций.

Функция `plot` имеет различные формы, связанные с входными параметрами, например `plot(y)` создает кусочно-линейный график зависимости элементов `y` от их индексов. Если в качестве аргументов заданы два вектора, то `plot(x,y)` создаст график зависимости `y` от `x`. Например, для построения графика функции `sin` в интервале от 0 до 2π, сделаем следующее

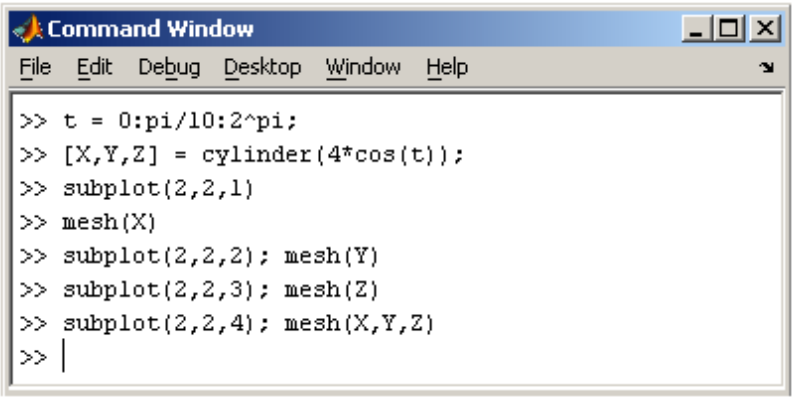


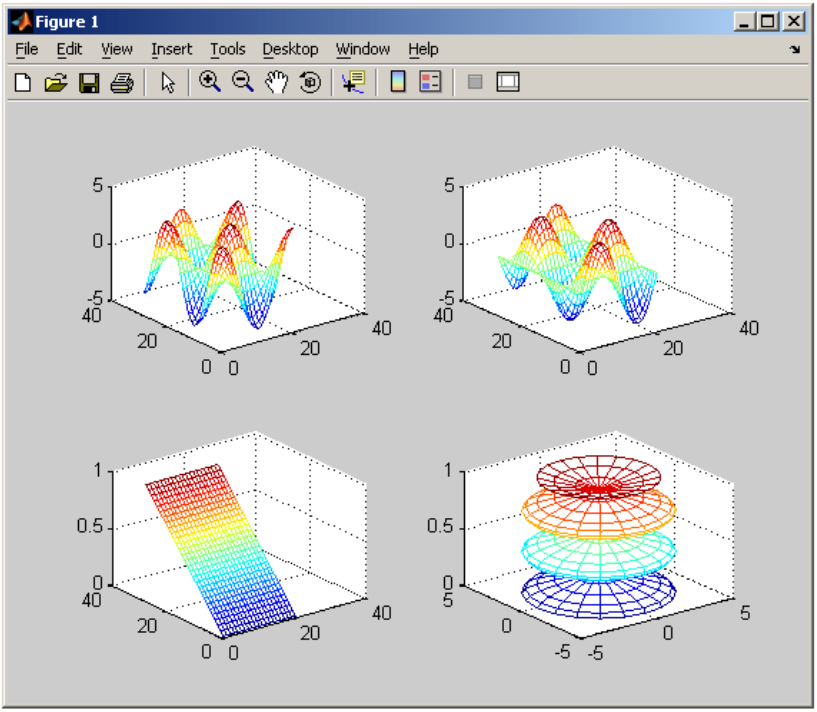
Программа построила график зависимости, который отображается в окне **`Figure 1`**



MatLab автоматически присваивает каждому графику свой цвет (исключая случаи, когда это делает пользователь), что позволяет различать наборы данных.

Команда `hold on` позволяет добавлять кривые на существующий график. Функция `subplot` позволяет выводить множество графиков в одном окне





4.5 Печать графиков

Пункт **Print** в меню **File** и команда print печатают графику MatLab. Меню **Print** вызывает диалоговое окно, которое позволяет выбирать общие стандартные варианты печати. Команда print обеспечивает большую гибкость при выводе выходных данных и позволяет контролировать печать из М-файлов. Результат может быть послан прямо на принтер, выбранный по умолчанию, или сохранен в заданном файле.

Список литературы

В.И.Горбаченко "Вычислительная линейная алгебра с примерами на MATLAB. СПб.: БХВ-Петербург, 2011. – 320с. (http://www.bhv.ru/books/full_contents.php?id=189162)

Шампайн Л. Ф., Гладвел И., Томпсон С. Решение обыкновенных дифференциальных уравнений с использованием MATLAB: Учебное пособие. 1-е изд. СПб.: Лань, 2009, 304 с.

В.Ф. Худяков, В.А. Хабузов Моделирование источников вторичного электропитания в среде MATLAB 7.x: учебное пособие. СПб.: ГУАП, 2008, 382 с.

И.Ф. Цисарь, В.Г. Нейман. Компьютерное моделирование экономики. М.: ДИАЛОГ-МИФИ, 2008.-384с.

И.Ф. Цисарь. MATLAB Simulink. Компьютерное моделирование экономики. М.: Солон-Пресс, 2008.-256с.

Чарльз Генри Эдвардс , Дэвид Э. Пенни. Дифференциальные уравнения и краевые задачи: моделирование и вычисление с помощью Mathematica, Maple и MATLAB. 3-е издание. Киев.: Диалектика-Вильямс, 2007. ISBN 978-5-8459-1166-7.

И. Черных. Моделирование электротехнических устройств в MATLAB, SimPowerSystems и Simulink. М.: ИД Питер, 2007, 288 с. ISBN 978-5-388-00020-0.

С.П. Иглин Теория вероятностей и математическая статистика на базе MATLAB. Харьков: НТУ "ХПИ", 2006,-612 стр.

С.Д.Штовба Проектирование нечетких систем средствами MATLAB. М.: Горячая Линия - Телеком, 2007,-288 стр. ISBN: 5-93517-359-X.